BENCHMARKING OF BARE METAL VIRTUALIZATION PLATFORMS ON COMMODITY HARDWARE

Duarte Pousa[*], José Rufino[*†]
[*]*Polytechnic Institute of Bragança, 5300-253 Bragança, Portugal*
[†]*Laboratory of Instrumentation and Experimental Particle Physics, University of Minho, 4710-057 Braga, Portugal*
(pousaduarte@gmail.com, rufino@ipb.pt)

ABSTRACT

In recent years, System Virtualization became a fundamental IT tool, whether it is type-2/hosted virtualization, mostly exploited by end-users in their personal computers, or type-1/bare metal, well established in IT departments and thoroughly used in modern datacenters as the very foundation of cloud computing. Though bare metal virtualization is meant to be deployed on server-grade hardware (for performance, stability and reliability reasons), properly configured desktop-class systems are often used as virtualization "servers", due to their attractive performance/cost ratio. This paper presents the results of a study conducted on such systems, about the performance of Windows 10 and Ubuntu Server 16.04 guests, when deployed in what we believe are the type-1 platforms most in use today: VMware ESXi, Citrix XenServer, Microsoft Hyper-V, and KVM-based (represented by oVirt and Proxmox). Performance is measured using three synthetic benchmarks: PassMark for Windows, UnixBench for Ubuntu Server, and the cross-platform Flexible I/O Tester. The benchmarks results may be used to choose the most adequate type-1 platform (performance-wise), depending on guest OS, its performance requisites (CPU-bound, IO-bound, or balanced) and its storage type (local/remote) used.

KEYWORDS

Bare Metal Virtualization, Synthetic Benchmarking, Performance Assessment, Commodity Hardware

1. INTRODUCTION

System Virtualization allows a physical host to run (simultaneously) many guest operating systems in self-contained software-defined virtual machines, sharing the host hardware, under control of a hypervisor. It comes in two basic forms: a) in hosted (or type-2) virtualization, an operating system (typically desktop-oriented) hosts virtual machines, along with other applications; this is achieved by type-2 platforms, like VMware Workstation (Vmware-a, 2016) or Oracle VM VirtualBox (VirtualBox, 2016); b) in bare metal (or type-1) virtualization, the platform hypervisor has direct hardware access, ensuring a more efficient use of the underlying resources; an approach is to install a thin hypervisor in the host, along with a management console; this console may be minimal, as in VMware ESXi (Vmware-b, 2016), or a management virtual machine, as in XenServer (XenServer, 2016) or Hyper-V (Hyper-V, 2016); another option is to have a server-class operating system with kernel modules that convert the operating system into a type-1 platform, like in KVM-based (KVM, 2017) approaches, as oVirt (oVirt, 2017) and Proxmox (Proxmox, 2017).

Bare metal virtualization is commonly deployed on server-grade hardware, often vendor-certified for specific type-1 virtualization platforms. This ensures adequate performance, stability and reliability, along with advanced management features, that are essential for mission-critical applications. However, it is usually possible to run most modern type-1 platforms on commodity hardware, and the compromises entailed by such option are acceptable in many scenarios, often related to tight budgets (home labs, public schools, SMEs, etc.).

This paper presents the results of several benchmarks conducted on Windows 10 LTSB and Linux Ubuntu Server 16.04 virtual machines, hosted in well-known bare metal platforms (VMware ESXi, Citrix XenServer, oVirt, Proxmox and Microsoft Hyper-V), running in commodity hardware with reasonable (tough not high-end) configurations. The benchmarks selected for this study include synthetic OS-specific benchmarks (Passmark for Windows, and UnixBench for Linux), and also a cross-platform I/O-centric benchmark (Flexible IO Tester). The later was run with virtual machines deployed both on local and remote storage, and several remote storage options were evaluated (NFS and iSCSI). The same benchmarks were conducted in native instances of the operating systems selected for this study, installed on the same host hardware.

All these test scenarios allowed to gather information that may help answering performance-centric questions, like: what is (are) the most suitable type-1 platforms(s) to host modern Windows and Linux guests in commodity "servers" ? what is the impact on secondary storage IO performance of different guest storage platforms ? what is the impact introduced by each type-1 hypervisor, compared to a native scenario ?

The rest of the paper is organized as follows. Section 2 describes our testbed, including type-1 platforms, guest operating systems and benchmarks selected, as well as hosts, guests and remote storage configuration. Section 3 presents the benchmarking methodology and its results. Section 4 revises related work on benchmarking of virtualization platforms. Section 5 concludes and points directions for future work.

2. TESTBED

2.1 Type-1 Virtualization Platforms

The bare metal virtualization platforms selected for this study range from proprietary and closed source, represented by VMware ESXi 6.0 (Free License) and Hyper-V (Windows Server 2016 Datacenter Edition), to open source implementations: Citrix XenServer 7.0, oVirt 4.1 and Proxmox 4.4. The specific versions of each type-1 platform were the latest ones available when this study began (fall 2016). Bearing in mind the distinction between a virtualization platform and a hypervisor, each of these platforms build on the following hypervisors, respectively: ESXi, Hyper-V, Xen and QEMU/KVM (for both oVirt and Proxmox). These hypervisors rely on different virtualization techniques, some used in combination (Hwang, J. et al, 2013): i) Para-Virtualization (Hyper-V, Xen, KVM); ii) Hardware-Assisted Virtualization, based on ISA extensions like Intel's VT-x and AMD-V (all hypervisors); iii) Full Virtualization (ESXi, Xen, KVM). In our study, the type-1 hypervisors choose automatically the virtualization technique to run each guest. Also, from this point onwards, the concepts of type-1 platform and type-1 hypervisor will be used interchangeably.

2.2 Guest Operating Systems

We conducted benchmarks on a small, yet representative set of guest operating systems (OSs): Microsoft Windows 10 (in its LTSB variant) and Ubuntu Server 16.04 LTS. Both choices play a dual role in this study.

The first OS stands as a modern representative of the Microsoft Windows line of operating systems and is poised to take the leading position in desktop machines. Windows 10 LTSB (Long Term Servicing Branch) is a long-term support version of Windows 10 Enterprise. It is designed for stability, meaning it will seldom (or ever) be updated with new features, but will have security updates and bug fixes for 10 years after its release.

The second OS selected for our study functions as a "proxy" for the many variants of Linux-based OSs, mostly used in infrastructure servers. Canonical's Ubuntu Linux distributions, specifically, have been consistently present at the top of usage rankings. Also, Ubuntu's relevance in the cloud space has been steadily growing, becoming a major player, both at infrastructure layers (IaaS) and as an instance (virtual machine). As with the LTSB Windows variant, a Long Term Support (LTS) Ubuntu version was also selected.

2.3 Benchmarks

Our original goal was to use only multi-platform, free (if possible open-source), and broad scope (i.e., not focused on a single system component) synthetic benchmarks. However, considering the choice of Operating Systems (see above), we were unable to find such a benchmark. Instead, we resorted to PassMark Performance Test v9 (PassMark-a, 2016) for Windows, and UnixBench v5.1.3 (UnixBench, 2016) for Linux. These are not multi-platform (and PassMark is not open-source), but all other requisites are fulfilled.

PassMark generates CPU, 2D graphics, 3D graphics, Disk and Memory ratings, that are combined in a global PassMark score (PassMark-b, 2017). UnixBench is a system (OS-level) benchmark for Unix-like systems, that mostly depends on the OS, libraries and compilers used. It includes tests on General

CPU+Memory performance (Dhrystone2), CPU Arithmetic performance (Whetstone), File Copying, Process Creation & Substitution, Context Switching, System Call Overhead, Shell Scripting and 2D/3D Graphics (unfit to Ubuntu Server and so excluded in our study). An overall system index combines the scores of all tests.

We still considered important to conduct a multi-platform benchmark, even if one component-specific, as a way to establish a comparable baseline between different operating systems executed on the same type-1 virtual hardware. This is accomplished by the Flexible I/O Tester (FIO, 2016) benchmark that was used to assess the impact of different storage options (local disk-based, remote NFS-based, and remote iSCSI-based).

2.4 Host, Guests and Storage Server Specifications

All benchmarks were executed on the same host, with the following configuration: CPU = Intel Skylake i7-6700 (4cores/8threads, 3.4/3.8 GHz); RAM = 32 GB 2133MHz DDR4; OS / Hypervisor Disk = 500 GB SSD; FIO Benchmark Disk = 500 GB SSD (capped to 32GB); Discrete GPU = NVIDIA Quadro 4000M.

Virtual guests were all configured with the same generic virtual hardware, as follows: vCPU = 1 socket, 4 cores/4threads; vRAM = 8 GB; OS vDisk = 32 GB thin provisioned (on top of the Hypervisor SSD); FIO Benchmark local vDisk = 32 GB thick provisioned (on top of the FIO Benchmark Disk); FIO Benchmark remote vDisk = 32 GB ZFS volume (on top of a remote SSD); Passthrough GPU = NVIDIA Quadro 4000M.

To support FIO benchmarking on the virtual guests with remote storage (via NFS and iSCSI), an auxiliary storage server was setup, with the following specifications: CPU = Intel Lynnfield i7-870 (4cores/8threads, 2.9/3.6 GHz); RAM = 16 GB 1333MHz DDR3; OS and Filesystem = FreeNAS v9.10 with ZFS; OS Disk = 16 GB USB3 PenDrive; Auxiliary Disk = 1TB 7200rpm SATA HD; NFS/iSCSI Disk = 256 GB SSD.

To make native and virtual conditions as close as possible, native tests ran with hyperthreading disabled (thus effectively ensuring that only 4 single-threaded cores were available), with RAM limited to 8GB, and using only 32GB of raw space from a separate SSD volume for FIO benchmarks. During guests execution, hyperthreading was enabled in the host system (as is usually done in virtual machine servers), although each guest used only 4 vCores. Also, there was no vCPU binding to real CPU cores. The default types provided by each hypervisor for vCPUs, vDisks and vNICs were used, without hypervisor optimizations or tuning.

All physical machines were connected in an isolated network, through a dedicated 1Gbps Ethernet switch.

3. EVALUATION

3.1 Methodology

Preliminary tests showed some variability on the scores achieved by very specific sub-benchmarks: DiskMark in the case of Passmark, and (Pipe-based) Context Switching in the case of UnixBench, with a Relative Standard Deviation (RSD) up to 7% and 25%, respectively (all other had RSD values below 3%). At the same time, a full run of Passmark and UnixBench complete relatively quickly in our testbed (around 30 minutes and 60 minutes, respectively). Thus, to ensure reliable results, it was decided to execute PassMark and UnixBench six times, on each testing scenario, with the outcome being averages of the six full runs. On the other hand, the time required for a full run of the FIO benchmark was much higher (from 6 to 12 hours, depending on the kind of secondary storage used), but each run produced very similar results (RSD values below 3.25%). Therefore, the outcomes of this benchmark are averages of only three runs.

Other testing conditions, common to all benchmarks, were as follows: i) each full benchmark run was preceded by a OS clean boot, followed by a five minute standby time to reduce any potential noise introduced by the booting phase; ii) before the first and last run, benchmarks conducted on virtual guests also implied a clean boot of the hypervisor (also followed by a five minute stabilization phase); iii) automatic update services were turned off in both operating systems tested, to prevent interference during testing; iv) finally, it should be stated that all benchmarks were run with their own default configurations.

Also, it's worth mentioning that, for each hypervisor, there are usually special drivers that are meant to be installed in guests in order to improve guest's performance at many levels (secondary-storage, 2D graphics, network, etc.). These correspond to VMtools in ESXi, XenTools in XenServer, Virtio drives for KVM-based hypervisors (like oVirt and Proxmox), and Integration Services for Hyper-V. Moreover, for Windows guests, VMware provides an optimization tool (VMware-c, 2016) that allows to disable unnecessary services and features to improve performance, and may be applied to Windows guests regardless of the underlying hypervisor. We have conducted benchmarks with/without the accelerations drivers (both in Windows and Linux guests), and with/without the optimizations offered by the optimization tool (Windows guests only). This paper focus on the results with the accelerations drivers installed, once we believe this is the most common scenario. Also, the effects of VMware's optimization tool are not reported here.

3.2 Results

Figures 1 to 3 provide charts with the consolidated results of the benchmarks conducted in our testbed. Columns represent percentages, relative to absolute scores measured in a native instance. Absolute scores are shown (in parentheses) on the top of the charts (in Figures 1 and 2), or below their vertical axis 100% mark (in Figure 3).

As expected, in general all hypervisors introduce performance penalties, to different extents, depending on the specific benchmark. However, we have observed some situations where virtual instances managed to be faster than the baseline native instance (this was especially true and most visible in UnixBench). Although rarely observed, this behavior is not entirely uncommon. In the right circumstances, type-1 hypervisors may provide faster than native performance, because they may be able to manage hardware resources more efficiently than the guest's kernel (Che, 2008; Henderson, 2009; Vaughan-Nichols, 2013).

3.2.1 PassMark results

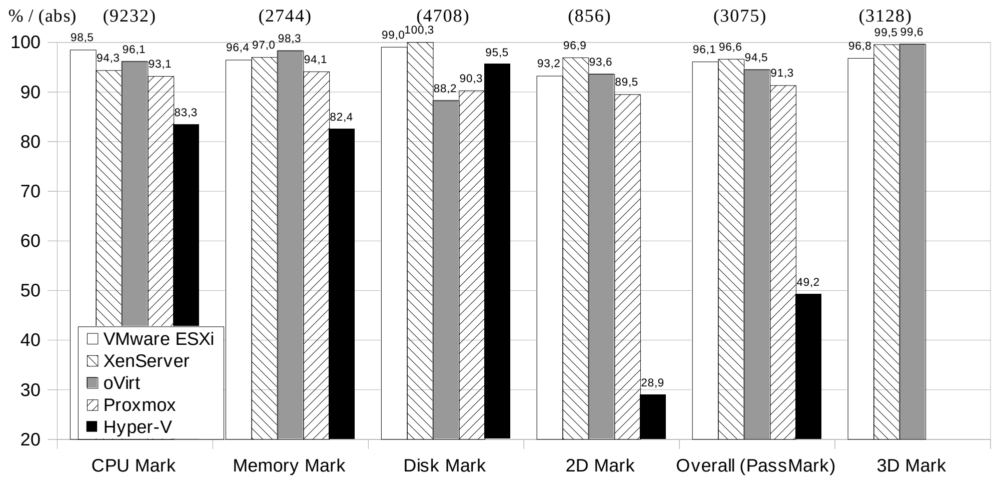

Figure 1. PassMark results (relative to native) for the Windows 10 LTSB guest (all hypervisors)

Figure 1 shows the results of PassMark in Windows 10 LTSB. In the Overall score, VMware ESXi, XenServer and oVirt are on the same league, with relative scores of 96.1%, 96.6% and 94.5% of the absolute native score (3075), whereas Proxmox is right below, with 91.3%, and Hyper-V achieves only 49.2%. This overall ranking is similar to the rankings in some PassMarks's sub-benchmarks, but visibly changes in others. We note that Overall (PassMark) results exclude 3D Mark results because GPU passthrough was not always entirely successful: in Proxmox, the guest gets stuck at 100% CPU utilization during boot, and in Hyper-V the guest halts during GPU compute tests. Thus, 3D Mark results are presented separately, and only for the hypervisors where GPU passthrough to the Windows 10 guest worked flawlessly and 3DMark completed.

In CPUMark, VMware ESXi and oVirt are the top performers, with 98.5% and 96.1% of the absolute native score (9232), thus becoming recommend choices for CPU-bound applications in a Windows 10 guest. XenServer and Proxmox closely follow, with 94.3% and 93.1%, respectively. Hyper-V achieves just 83.3%.

MemoryMark results are relatively homogeneous: except for Hyper-V, all scores are within a margin of 4.2% of each other, between 94.1% and 98.3% of the native score (2744). Hyper-V, lags behind, to 82.4%.

However, Hyper-V exhibits a good performance in DiskMark, attaining 95.5% of the native score (4708), yet still surpassed by ESXi and XenServer, with 99% and 100.3%, respectively, thus with virtually no performance loss. In turn, oVirt and Proxmox, fall to 88.2% and 90.3%, respectively; this may be due to the default Virtio configuration used: it assumes a "native" instead of a "threads" IO mode, meaning no additional

IO threads are created by QEMU to handle extra IO load. As such, for Disk-bound applications with IO load similar to DiskMark, running in a Windows 10 guest, instantiated on local hypervisor storage, ESXi and XenServer ensure the very best (if not optimal) performance, with Hyper-V right behind.

Regarding graphical performance, XenServer leads in 2DMark, with 96.9% of the native score (856), slightly above ESXi (with 93.2%), oVirt (with 93.6%) and Proxmox (with 89.5%). Hyper-V, however, shows very poor performance, with a relative score of only 28.9%, that brings down its Overall (PassMark) score.

3DMark, that ran separately, achieved close-to-metal performance in XenServer and oVirt, with a relative score of 99.5% and 99.6% of the native score (3128), respectively. ESXi, with 96.8%, is near. Were 3DMark included on the Overall score, it would confirm VMware ESXi, XenServer and oVirt as the top hypervisors.

3.2.2 UnixBench results

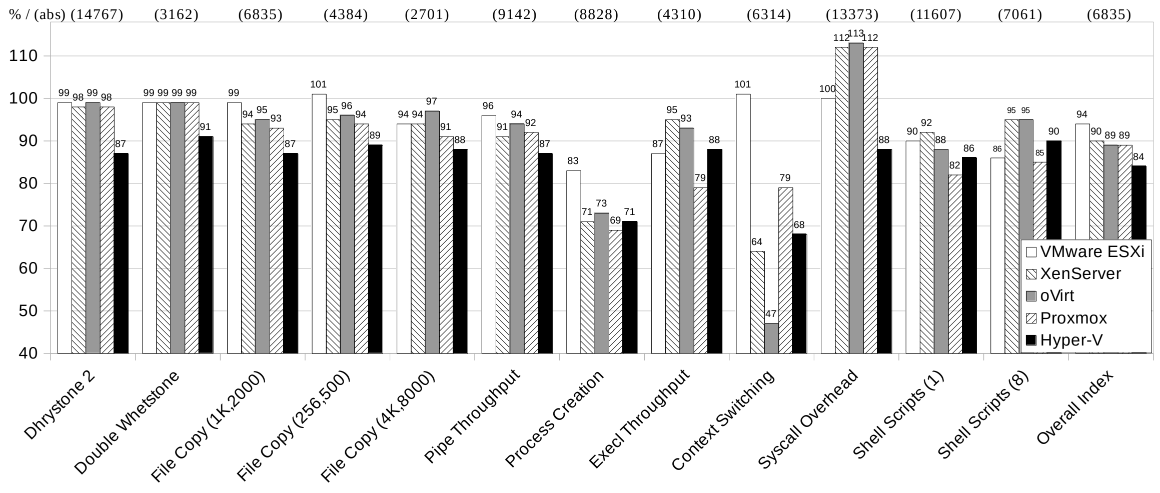

Figure 2. UnixBench results (relative to native) for the Ubuntu Server 16.04 LTS guest (all hypervisors)

Figure 2 presents the results of UnixBench in Ubuntu Server 16.04 LTS. As stated before, UnixBench is an OS-level benchmark, not meant to evaluate (at least directly) hardware components. It generates single and multi-threaded results, and we only provide the later, meaning each test ran in parallel, in 4 separate instances, one per each real/virtual CPU-core. Also, results are presented truncated (integer part only) due to space limits.

In the Overall Index, VMware ESXi leads with 94% of the native index (6835), closely followed by XenServer (90%), and both oVirt and Proxmox (89%), all very near to each other. Hyper-V attains only 84%. Thus, running Linux guests seems a task for which Hyper-V is not as suitable as the other type-1 hypervisors.

In the CPU-related/bound tests (Dhrystone and Whetstone), VMware ESXi, XenServer, oVirt and Proxmox achieve near native performance: between 98% and 99% of the native indexes (14767 and 3162). The dominance of these hypervisors in such tests is in line with what was observed in the CPU Mark test. The last rank of Hyper-V in CPU Mark is also transposed to Dhrystone and Whetstone, with only 87% and 91%.

File Copy tests measures "the rate at which data can be transferred from one file to another" (UnixBench, 2016), using various buffer sizes (256, 1024, 4096 bytes) and file sizes (2000, 500, 8000 blocks). They provide clues on file system performance and, indirectly, on secondary storage performance. Taking the average of the relative indexes in these tests, the following hypervisor ranking emerges: VMware ESXi – 97%; oVirt – 95.6%; XenServer – 93.3%; Proxmox – 92.3%; Hyper-V – 88%. For what it's worth, this ranking is only consistent with DiskMark's in VMware ESXi and Proxmox; the other hypervisors exchange rankings.

Pipe Throughput measures the rate at which the same process can write data to a pipe and read it back. This "has no real counterpart in real-world programming" (UnixBench, 2016), and we only provide results for completeness. Much in line with other tests, VMware ESXi is first, and Hyper-V is again the last.

Process Creation, that measures the rate at which a process can fork and reap a child that exits right away, exhibits relatively low scores: VMware ESXi achieves 83% and the other hypervisors fall between 69% and 83%. This test is said to be related to memory bandwidth, due to memory allocations for new processes (UnixBench, 2016). Yet, there's no obvious relation with MemoryMark results, perhaps because of the light nature of the Process Creation benchmark: children die immediately, without changing any variables, and with the parent waiting for their ending; thus, the parent's data segment never gets duplicated, and so, basically, a child needs only a stack and kernel memory for its process control block (the code segment is always shared).

Execl Throughput measures the rate at which a process image (code and data) in RAM may be replaced. This involves reading the new code and data from an executable at the file system (FS), into a memory region. However, in this benchmark, the same (small) executable replaces itself, and so FS caching comes into play. We dare to say that this benchmark ends up evaluating memory bandwidth, and some hypervisors, namely oVirt, Proxmox and Hyper-V, even keep their relative positions from the Process Creation test: oVirt first, then Hyper-V, then Proxmox (VMware ESXi and XenServer, however, exchange places). If we take the average of the Process Creation and Execl Throughput benchmarks, the following ranking emerges: VMware ESXi – 85%; XenServer – 83%; oVirt – 83%; Hyper-V – 79,5%; Proxmox – 74%. These values are more balanced, thus more in line with the Memory Mark scores, although lower by more than 10%.

The Context Switching test measures the rate at which two processes exchange an increasing integer through a pipe. Because both processes strictly alternate execution, this test assesses how fast is context switching (an OS capability important to many real-world applications). It turns out that the results of this test are the most divergent in all UnixBench tests, with a RSD of 22,7%. VMware ESXi is able to sustain faster than native performance, up to 101% of the native index (6314), and Proxmox follows at distance, with 79% score. However, XenServer, Proxmox and Hyper-V only achieve 64%, 47% and 68%, respectively. The oVirt score is intriguing, and deserves further investigation, mostly because, like Proxmox, is KVM-based.

In the Syscall Overhead test, a basic primitive (the `getpid` system call, that returns the ID of the calling process) is repeatedly invoked, so as to measure the cost of entering and exiting the kernel. This time, all hypervisors but Hyper-V exhibit exceptionally good performance, from 100% of the native index (13373) in VMware ESXi, up to 112% / 113% in XenServer and KVM-based hypervisors. Thus, a syscall acceleration mechanism seems to come into play, in the three Linux-based hypervisors. Hyper-V falls behind, at 88%.

Finally, the Shell Scripts (*n*) tests measure the rate at which a process can start and reap a set of *c*n* concurrent copies of a script that applies a series of transformation to a data file, where *c* is the number of CPU cores seen by the OS. Once *c*=4 in our testbed, Shell Scripts (1) and Shell Scripts (8) reach up to 4 and 32 concurrent instances of the child script. Comparing the results from both scenarios, XenServer, oVirt and even Hyper-V, react more favorably to a load increase, hinting at a better scalability under higher load (this is subject to confirmation in future work, when performing benchmarks with more than one virtual machine running).

3.2.3 FIO results

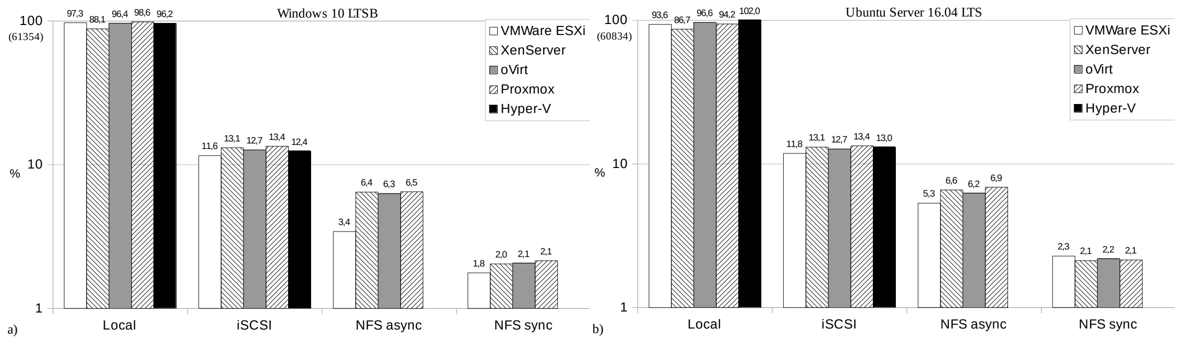

Figure 3. FIO results (relative to native) for a) Windows 10 LTSB, and b) Ubuntu Server 16.04 LTS (all hypervisors)

The charts of Figures 3.a) and 3.b) exhibit the results of FIO in Windows 10 LTSB and Ubuntu Server 16.04 LTS, respectively. FIO's native absolute scores are presented below the chart's vertical axis 100% mark.

Native scores on both OSs are very similar (with a relative difference of ≈ 0,0085%); as such, relative scores in both charts are directly comparable, allowing to confront both OSs for any kind of storage used.

An immediate conclusion is that FIO performance in both OSs is very similar (excluding VMware ESXi over NFS). This supports the following cross-platform conclusions: local SATA SSD storage is ≈10 times faster than the fastest remote storage, provided by the iSCSI protocol; using an NFS share with asynchronous writes (enforced at the ZFS level, on the storage server), ensures approximately half of the performance of an iSCSI volume; finally, NFS with synchronous writes (the default NFS behavior) is roughly one third as fast as asynchronous NFS. We note that results are not provided for Hyper-V and NFS because Hyper-V does not currently support (and likely won't) virtual disks instantiated on NFS shares (Kolomyeytsev, 2015).

Trying to discover which hypervisor(s) offer the best FIO performance, for each kind of storage, we observe that: i) on local storage, all hypervisors provide very good performance (96,2% to 98,6% in Windows, and 93,6% to 102% in Ubuntu), except XenServer (with only 88,1% in Windows, and 86,7%, in Ubuntu); notably, this contradicts XenServer's DiskMark score of 100,3%, above all others; however, XenServer's FIO scores are more in line with UnixBench's File Copy scores, where XenServer was undercut almost just by Hyper-V; ii) with iSCSI, results are very balanced across all hypervisors, and strikingly similar between both OSs; iii) with NFS, results are also balanced, excluding the score produced by the Windows guest on VMware ESXi over NFS asynchronous, that is about half of the other hypervisors.

4. RELATED WORK

We only reference here academic work, that conducted performance studies focused on type-1 hypervisors.

Che et al. (Che, 2008) studied the performance of the Xen 3.1 and KVM-60 hypervisors (supported by a Fedora 8 host), when running the Linpack, LMbench and IOzone benchmarks, to measure floating-point CPU power, OS latencies and memory bandwidth, and file system I/O performance, respectively, on Linux Fedora 8 and Windows XP guests (Linpack only). The physical host was a desktop-class machine, with a dual-core CPU and 2 GB of RAM (1.5 GB for guests). Xen ensured close-to-native Linpack and LMbench performance, and sometimes better than native IOzone performance.

Fayyad-Kazan et al. (Fayyad-Kazan, 2013) conducted a study with Linux instances modified with RT-Preempt realtime preemption patches, on VMware ESXi 5.1, Xen 4.2.1, Hyper-V 2012 and bare metal. The host was workstation/server-class (Xeon CPU), but without hyperthreading capabilities. Guests had only 1 vCPU and 1GB of RAM, a rather limited configuration, but some tests were made with more than one VM running, contrarily to our study, that runs a singles VM at a time. The benchmarks were tailored to evaluate the real-time performance and behavior of real-time OSs (which is a corner case). Xen performance came up very close to native, followed by Hyper-V and ESXi.

Hwang et al. (Hwang, 2013) evaluated the performance of Hyper-V 2008, KVM, VMware ESXi 5.0 and Xen 4.1.2 under hardware-assisted virtualization settings, using microbenchmarks for CPU (Bytemark), memory (Ramspeed), disk (Bonnie++ and FileBench), and network (Netperf), and also application level benchmarks (Linux kernel compilation and freebench). They also ran several VMs and studied their mutual interference. The host used was server-class, with a four core Xeon CPU, 8 GB of RAM, SAS hard disks and 1 Gbps Ethernet NICs. Guests were based on Ubuntu 10.04 LTS, with 1 or 4 vCPUs, 2 GB of RAM for isolated VMs, and 1GB of RAM for simultaneous VMs. Results showed that, overall, VMware ESXi performed the best, followed by a respectable performance of the other three hypervisors.

Graniszewski et al. (Graniszewski, 2016) analyzed the performance of Hyper-V 2008+2012, VMware ESXi 5.1, Oracle VM Server, XenServer 6.1, and Oracle VM Virtualbox, using component-specific benchmarks (nbench for CPU, netperf for network, FileBench for storage, and ramspeed for memory), that ran on an Ubuntu 12.04 guest. A real-word test (Linux kernel compilation) was also performed. Tests were done with one and two vCPUs, and 2GB RAM. The host was desktop-class, with a low-end configuration (dual-core CPU, 4GB RAM, 5400rpm HD). Results put VMware ESXi on the lead, followed by Xen.

5. CONCLUSIONS AND FUTURE WORK

As discussed throughout this paper, the performance characteristics of all the mainstream type-1 virtualization platforms were analyzed, through synthetic benchmarks conducted in guests, with care to avoid resource over-commitment (thus limited to a single guest in operation). The scope of the benchmarks used enabled investigation under multiple workloads aiming at stressing CPU, Memory, local and remote storage, GPU (under passthrough) or even OS (Linux) sub-systems. Ultimately, all the tested type-1 platforms proved that, in the right conditions, there is very little overhead introduced by the hypervisors, with the exception of Hyper-V, the most frequent outlier in our tests. The choice of a type-1 platform, though, is not purely dictated by raw performance, especially when margins between different options are narrow. Other factors, like the technical background of the IT staff, the organization culture, and the IT budget, play perhaps a more decisive role. But if we were to recommend a type-1 virtualization platform, based only on the performance results observed (and their consistency), VMWare ESXi and Citrix XenServer would be our current choices.

While the provided analysis is thorough, there are still many untested scenarios that we intend to cover in future work, such as: i) evaluate the performance of (para)virtualized/SR-IOV network adapters; ii) introduce 10Gigabit Ethernet into the testbed (following the market trend that points to its commoditization) and study its effect on network and remote storage (FIO) benchmarks; iii) go beyond the default configurations for each hypervisor, trying to assess the best combination(s) of critical setting(s) (CPU types, Main Memory paging, IO options); iv) conduct a representative set of real world benchmarks (including Linux kernel compilation, video transcoding, etc.); v) assess the performance in multi-server, multi-VM and over-commitment scenarios, taking advantage of industry standard benchmarks, namely SPEC VIRT_SC 2013 (SPECvirt, 2013); vi) study and compare the impact of features like failover / high-availability support, load balancing, live migration, etc.

REFERENCES

Che, J. et al, 2008. Performance Measuring and Comparing of Virtual Machine Monitors. Proceedings of 2008 IEEE International Conference on Embedded and Ubiquitous Computing. Shanghai, China, pp 381-386.

Fayyad-Kazan, H. et al, 2013. Benchmarking the Performance of Microsoft Hyper-V server, VMware ESXi and Xen Hypervisors. In Journal of Emerging Trends in Computing and Information Sciences, Vol. 4, No. 12, pp 922-933.

FIO, 2016. Flexible I/O Tester. Available at: https://github.com/axboe/fio (Linux source code), http://bluestop.org/fio/ (Windows binaries), (Accessed: October 2016)

Graniszewski, W. and Arciszewski, A., 2016. Performance analysis of selected hypervisors. In International Journal of Electronics and Telecommunications, Vol. 62, No. 3, pp 231-236.

Henderson, T. et al, 2009. Xen-based hypervisors push performance limits. Available at: http://www.networkworld.com/article/2271464/data-center/xen-based-hypervisors-push-performance-limits.html (Accessed: June 2017)

Hwang, J. et al, 2013. A component-based performance comparison of four hypervisors. Proceedings of 2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013). Ghent, Belgium, pp 269-276.

Hyper-V, 2016. Try Active Microsoft Hyper-V Server 2016 | TechNet Evaluation Software. Available at: https://www.microsoft.com/en-us/evalcenter/evaluate-hyper-v-server-2016 (Accessed: October 2016)

Kolomyeytsev, A., 2015. Hyper-V: NFS. Available at: https://www.starwindsoftware.com/blog/hyper-v-vms-on-nfs-share-why-hasnt-anyone-thought-of-that-earlier-they-did-in-fact-2 (Accessed: October 2016)

KVM, 2017. Kernel Virtual Machine. Available at: https://www.linux-kvm.org/page/Main_Page (Accessed: January 2017)

oVirt, 2017. oVirt. Available at: https://www.ovirt.org/ (Accessed: January 2017)

PassMark-a, 2016. PassMark PerformanceTest. Available at: http://www.passmark.com/products/pt.htm (Accessed: October 2016)

PassMark-b, 2017. PassMark Formulas. Available at: http://www.passmark.com/forum/performancetest/4599-formula-cpu-mark-memory-mark-and-disk-mark (Accessed: June 2017)

Proxmox, 2017. Server virtualization management with Proxmox VE. Available at: https://www.proxmox.com/en/proxmox-ve (Accessed: January 2017)

SPECvirt, 2013. SPEC virt_sc 2013. Available at: https://www.spec.org/virt_sc2013/ (Accessed: September 2017)

UnixBench, 2016. BYTE Unixbenchmak suite. Available at: https://github.com/kdlucas/byte-unixbench (Accessed: October 2016)

Vaughan-Nichols, J.S., 2013. Yes, virtualization is faster (sometimes) than native hardware. Available at: http://www.zdnet.com/article/yes-virtualization-is-faster-sometimes-than-native-hardware/ (Accessed: June 2017)

VirtualBox, 2016. Oracle VM VirtualBox. Available at: https://www.virtualbox.org (Accessed: October 2016)

VMware-a, 2016. Workstation for Windows – VMware Products. Available at: https://www.vmware.com/products/workstation.html (Accessed: October 2016)

VMware-b, 2016. Free VMware vSphere Hypervisor. Free Virtualization (ESXi). Available at: https://www.vmware.com/products/vsphere-hypervisor.html (Accessed: October 2016)

VMware-c, 2016. VMware OS Optimization Tool. Available at: https://labs.vmware.com/flings/vmware-os-optimization-tool (Accessed: October 2016)

XenServer, 2016. XenServer | Open Source Server Virtualization. Available at: https://xenserver.org (Accessed: October 2016)